International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 19 (2018) pp. 14457-14460

Interaction System using Kinect Sensor for Development of Human-Virtual Objects Game

Gankhuyag Ochirsum
Department of Game Engineering, Pai Chai University, Daejeon, Korea.

Young-Ho Sohn
Department of Computer Engineering, Yeungnam University, Gyeongsan, Korea.

Dong-Won Park*
Department of Game Engineering, Pai Chai University, Daejeon, Republic of Korea.

Abstract

This research targets development and implementation of the controller-less gaming interface on the basis of Kinect Sensor with Chroma Key. The purpose of this work is to enable users to try full performance of Windows 10 PC game set up with Kinect sensor, using just gestures and motions, and ignoring essential to use any physical devices such as mouse, keyboard or game controller. The main result of the research is one completely functioning game that you can play without any controller. Players can interact with virtual objects using gesture or motion which can be detected by Kinect Sensor based on human skeleton joints in real-time.

Keywords: Chroma Key, Kinect, Windows PC, Human skeleton joints

INTRODUCTION

Human motion and gesture analysis are getting more and more intentions in the human-computer interaction area. In one aspect, such a requirement is created by allowing the current device to be merged as more sophisticated and computational capabilities increase, thereby solving complex problems. On the other hand, recently cheaper devices are produced that can be used as part of a relatively inexpensive system. The main areas of human gesture analysis are supervision, film, game, human-computer interface and animation. The major fields of work are human motion analysis, gesture recognition and tracking.

Current research is primarily focused on human body recognition and focuses on developing games that can operate the Windows 10 in a touchless and clickless manner using the power of Kinect Sensor. For this reason, there is no physical interaction to the game controller or keyboard. Enabled visions of no physical interaction with external devices, it is focused to model mouse functionalities and detecting predefined motions and gestures.

KINECT SENSOR

Kinect sensor is a line of motion analyzing input devices that was produced by Microsoft for Xbox One video game consoles and Microsoft Windows PCs. Based around a web camera-style add-on external, it allows players to control and contact with their device/computer without the use for a game controller, over a natural user interface applying gestures and voiced commands. It applies a large-angle time-of-flight camera, and processes 2 gigabits of data per second to read its environment. The new Kinect sensor has more accuracy with 3 times the reliability over its ancestor and can recognize without clear light by using an active Infrared sensor. It has 60% larger area of vision that can recognize a player up to three feet from the Kinect sensor, compared to six feet for the initial Kinect sensor, and can detect up to six skeletons at once. It can also recognize a user`s facial expression, heart rate, the orientation and position of twenty five individual joints, the density put on each limb, speed of user motions, and detect motions performed with initial controller[1]. Kinect`s color camera records 1080p video which can be shown in the exact same resolution as the showing screen, enabling for a full range of scenarios. Also improving video computings and video analysis applications, this enables a durable input on which to create interactive applications. Kinect sensor`s microphone is used to implements voice commands for functions such as tracking, executing games, and waking the console from sleep mode.

Skeleton Tracking enables Kinect to detect users and track their actions. Using the infrared(IR) camera, Kinect sensor can detect up to 6 players in the field of view of the Kinect sensor. Of these, up to 2 players can be detected in detail[2][3]. An application can place the joints of the detected players in space and analyze their actions over time.

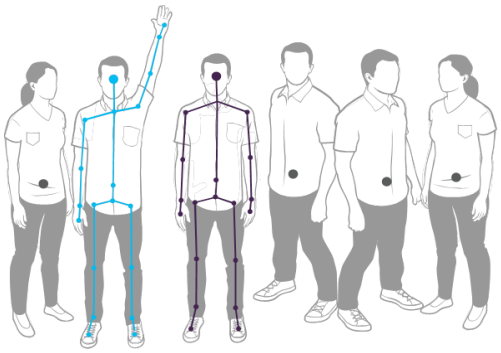

Figure 1. Kinect Sensor recognition with six people

Skeletal Tracking is designed to detect users sitting or standing, and facing the Kinect sensor sideway positions give some challenges about the part of the player that is invisible to

the Kinect sensor[4]. To be tracked, players simply have to be in front of the Kinect sensor, making sure the Kinect sensor can see user's upper body and head no special position or calibration action has to be taken for a player to be detected.

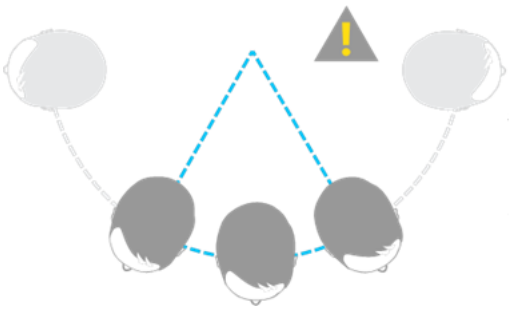

Figure 2. Skeleton tracking capability for user position

FIELD OF VIEW

Kinect sensor field of view of the players is detected by the configuration of the Infrared(IR) camera, which are the depth range enumeration type. In actual field mode, Kinect sensor can see users standing between 0.7 meters and 3.9 meters away. Players will need to be able to operate their limbs at that range, recommending a practical distance of 1.3 to 3.4 meters[5].

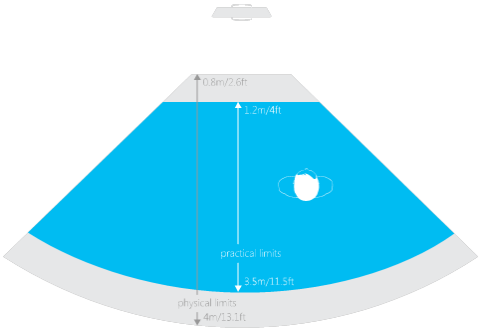

Figure 3. Kinect horizontal Field of View in default range

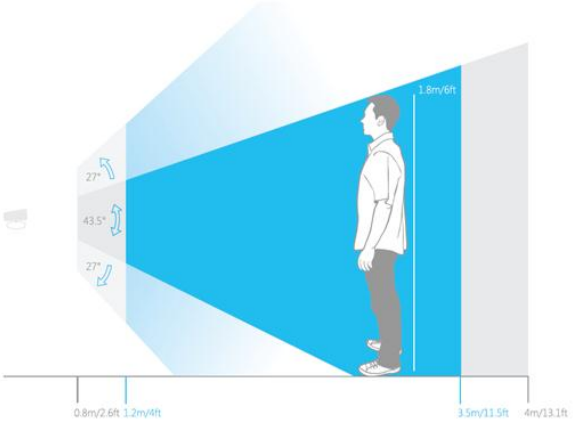

Figure 4. Kinect vertical Field of View in default range

Skeleton Position and Tracking State

The skeleton in a frame have can have a tracking state of "tracked" or "position only". A detected skeleton gives detailed information regarding the position in the Kinect camera`s range of view of 20 joints of the player`s body. A skeleton with a detecting state of "only position" has data regarding the pose of the player, but zero details about the joints. An application can decide which skeletons to track, using the tracking ID[6].

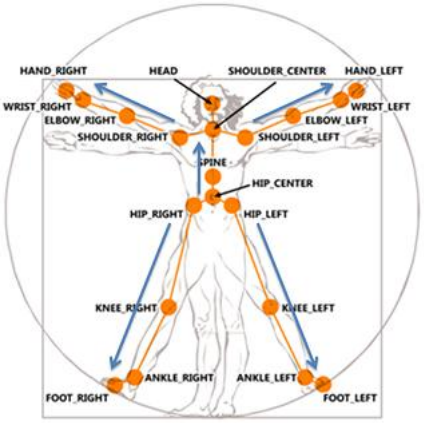

Figure 5. Skeleton joint indexes

Chroma Key Technique

There are several different quality and speed optimized techniques for implementing color keying in software. In most versions, a function f(r, g, b) -> α is applied to every pixel in the image. α(alpha) has a meaning similar to that in alpha compositing techniques. $\alpha \leq$ means the pixel is fully in the green screen, $\alpha \geq 1$ means the pixel is fully in the foreground object, and intermediate values indicate the pixel is partially covered by the foreground object(or it is transparent). A further function g(r, g, b) -> (r, g, b) is needed to remove green spill on the foreground objects[7].

A very straightforward g() function for blue screen is B(r + c) – Dg where B and D are player flexible constants with a default value of 2.0. A very simple f() is (r, min(f, c), c). This is similar to the capacities of analog and movie-based screen making[8].

Chroma Key for Kinect Data

When we refer to "Background removal", we need to keep the pixels which form the user and remove anything else that does not belong to the user. The depth camera of the Kinect sensor comes in handy for determining a user`s body. However, we need to find the RGB color values, not the depth distances. We need to specify which RGB values correspond to the user`s depth values. The depth camera gives us the depth value and the RGB camera provides us with the color value. We map those values using CoordinateMapper. CoordinateMapper is a useful Kinect property that determines which color values correspond to each depth distances.

Image processing is a major subfield of computer science and electrical engineering and to a lesser extent biomedical engineering. Graphics are usually represented via two methods: vector or bitmap. Vector graphics take a more abstract approach and use mathematical equations to represent lines, curves, polygons and their fill color. When a program opens the vector image it has to translate those equations and render the image. This vector approach is often used to represent clipart and 3D animations. Bitmap images take the opposite approach and simply represent the image as a 2D array of pixels. Each pixel is a small dot or square of color. The bitmap approach is used most commonly for pictures, video, and other images. In addition, bitmaps do not force us to translate the vector equations, and thus are simpler to manipulate for our purposes.

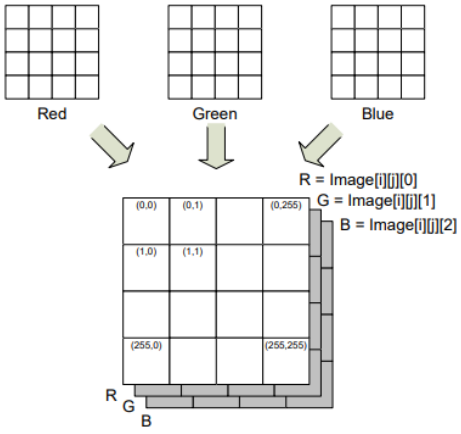

Figure 6. Representation of a color bitmap as a 3D array

The primary task in performing the chroma key operation is to determine which pixels of the image should be classified as the chroma key(green). Part of the problem is that shadows and other artifacts may make the background pixels significantly lighter or darker shades of the key color, especially the closer they are to the foreground image. We would like an algorithm that is robust enough to classify these outlier pixels correctly without incorrectly classifying pixels of the foreground image. Whatever method we use, the goal is to create a 2D "mask" array that marks foreground image pixels(say with a 1) vs chroma key pixels(with a 0). Once this mask is created, we can easily create the output image by using the mask to select pixels from the input or background image.

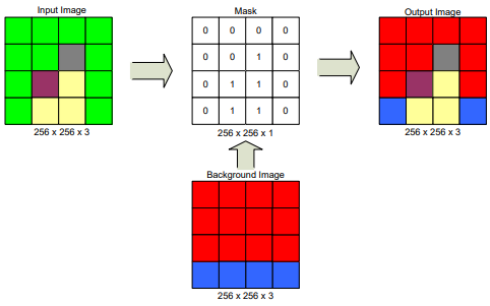

Figure 7. Chroma key process

Many methods could be devised for solving whether a pixel color is part of the chroma key. One may think of simply checking if a pixel`s green component is above a certain value. However, usually this does not work because light colors like white and gray are represented as the superposition of all three RGB colors.

A better approach may be to think of each pixel`s RGB components as a coordinate of a point in 3D space. Next, take some number of sample pixels that we would know to be the chroma key color(ones near the edges for example). We can average them to create a reference point that represents the average chroma key color[9]. Then any pixel within a certain distance from the reference can be considered part of the chroma key while pixels further away will be considered part of the foreground image. This method can work for any chroma key but does require some trial and error to set the threshold distance for which pixel values will be determined to be the chroma key or foreground image.

GAME RESULT

The player`s replicant on the screen shoots to virtual object and that virtual object will be destroyed when bullet reaches to object. Strategically place your body in order to avoid virtual object. Using your hand`s position aligned to the computer screen`s coordinate system to move the cursor in the position the user points on the screen. Gestures are recognized which activate specific computer mouse and keyboard commands when they performed by the user to interact Virtual Objects in game. The gesture with the greatest probability defines which game action will be activated. All the commands are categorized among their usage and they are consisted in separate Types of Game. The mouse commands such as the Left Click, the Right Click, The Double Click and the Grab and Ungrab commands are also used.

CONCLUSION

Natural user interfaces are becoming more part of our everyday life. It is important to find good designs for gestures that are well adapted to their purposes. The primary goal for NUI is to be efficient, but also to be simple and favorite for the majority of users. If users do not like the way they control, they will go somewhere else without adopting the system. The point of this work is the creation of controller-less games using Kinect Sensor with Chroma Key technique. Gestures have been designed and implemented. Each gesture is designed for one of the actions in game, the aim, the shoot and the select. And we also integrated the Chroma key technique with the game. Games with Chroma key technique have become more realistic and fun.

ACKNOWLEDGEMENT

This work was supported by the research grant of Pai Chai University in 2018. The corresponding author is Dong-Won Park.

REFERENCE

[1] Vardakis Evangelos. Gesture Based Human-Computer Interaction Using Kinect. Technological Educational Institute of Crete, March 2017.

[2] Ioanna Tziouvara. Integration of 3D tracking systems for Interaction in Spatial Augmented Reality. Delft University of Technology, December 14, 2012.

[3] Kinect. https://en.wikipedia.org/wiki/Kinect.

[4] Samet ERAP. Gesture Based PC Interface with Kinect Sensor. Tallinn University of Technology, 2012.

[5] Satish Kumar Thati, Venkata Praneeth M. Determining the Quality of Human Movement using Kinect Data. Blekinge Institute of Technology, June 2016.

[6] Chroma key. https://en.wikipedia.org/wiki/Chroma_key.

[7] Lin Yang ,3D Sensing and Tracking of Human Gait, University of Ottawa, 2015.

[8] Primatte chromakey technology. https://en.wikipedia.org/wiki/Primatte_chromakey_technology.

[9] Chroma Key. https://github.com/brianchirls/Seriously.js/wiki/Chroma-Key.